random number generator) if desired. Statistical comparisons using digits in the base 2 method and system also show better randomness than the first 5,000 digits of pi. See the frequency table at the end of this reply for testing results.

The Enablement requirement discussed on pages 2 and 3 of the Office Action document has been complied with. One skilled in the art could certainly make and use the invention proposed. In fact, the entire software algorithm was submitted as written in Visual Basic on microfiche as the USPTO requires. One skilled in that computer language can see exactly how the algorithm functions and reproduce it or modify it for experimentation. A disk of the working software is available upon request for installation in another PC. Instructions for using the software of the less than preferred based 10 version is available in the help menu when the program is running. Using it for encryption and decryption is very intuitive. A few simple instructions should suffice to easily encrypt and decrypt with the program if one wishes to avoid the help menu. The enablement requirement has been met. More explanation will be gladly provided and any questions answered.

In **Claim 1** the question was asked in the Office Action document on page 3: "What part of the applicant's system produces the improvement of producing c-text that varies when a message is coded even if said message is coded many times with the same key." The answer is that the one way function called EndPoint Permutation is used to generate pads to logically combine with plaintext forming a random appearing character string. In binary form this would typically involve using the XOR function, instead of Chinese

summing, to make the two strings of equal length into one. I have not found a single reference after a search in mathematics literature for this one way function used to accomplish permutation of prime and non-prime element strings. There is also no published reference to the one way function used in an encryption algorithm before, even though it is more secure than multiplying large primes together as Public Key encryption requires. The EndPoint Permutation and other unique aspects of the software program remain the same when applied to bits instead of digits. As described in the present invention the improved results also stem from interspersing the resulting bit string of c-text with control bits assigned to numbered positions as determined by the user key. When a cryptanalyst examines c-text they will not be able to determine what characters represent plain text after it has been XOR'd with a pad or whether the c-text character is a composite of control bits, XOR'd plain text, or bits that determine the program number used in the encryption block examined. Statistical analysis will also not help to resolve the true nature of the c-text characters. In the preferred binary version only a single character acting as an anchor to the c-text block is guaranteed to not represent bits with a combination of one to three different functions described. The anchor character is also used to reveal what bits the 64 possible c-text characters are in c-text. The first step of decryption is to strip off the anchor character to determine what bits each of the possible 64 c-text characters in the c-text represent. Then six more control bits are removed from numbered positions within the c-text bit string to determine the program number. Once the program number is known the remaining control bit positions are known and removed from the bit string. What remains is the XOR'd plaintext. The control bits are then used to reconstruct the pad used which can then removed to reveal pure plaintext bits.

Permutation of the plaintext bits is then reversed to restore the original order using the control bits.

All of the above described means contribute synergistically to achieve computationally secure communication.

Claims 1-19 that were rejected on page three of the Office Action document since it is called an "omnibus type claim" stating that "the applicant has not disclosed the method claimed for transformation and transposition." This is incorrect since the method of transformation and transposition has been described in text and in the computer language of Visual Basic. One skilled in the art can see from the encryption algorithm submitted and accompanying explanation exactly how the transformation and transposition is done. Using the software program should answer any remaining questions. Perhaps this will help anyone examining the proposed invention to see that a definite method for transposition and transformation is employed, otherwise the software would not work. It certainly would not work as well as it does as demonstrated from analysis of the randomness of the cipher text it produces. The transposition and transformation methods employed are not unlimited nor are they unidentified as one skilled in the art can see. The fact that transformation and transposition is performed in a myriad of ways is simply a reference to the fluidity of the algorithm made. A user can choose myriad user keys that affect how the transformation and transposition is performed. The number of programs within the algorithm can also be modified in a number of ways or similar programs may be added in future revisions of the software. A request to patent all the possible variations

is not requested just the unique claims achieving superior results and its obvious variants submitted.

Claims 1-9 and 12-19 rejections on page 4 of the Office Action document is addressed.

Claim 1 on page 5 of the Office Action document says that Yanovsky has already patented this claim. In reviewing his patent 5,703,948 there is no mention of anything resembling the EndPoint Permutation method described in the claim. There is no mention of using a prime number of digits or bits by Yanovsky to generate a random pad to disguise plain text. There is no mention of this specific superior one way function being used to harvest bits from a ring of bits, as determined by a user key, when circled. On page 11 of Yanovsky's patent he states plainly that "the dynamic random keys are changed using the random characteristics inherent in the messages themselves, and are changed internally within the two units in synchronism with each other, and wherein the synchronism is maintained during the transmission of messages, and from one message transmission to the next, without the need for transferring keys or for a master key." Yanovsky's method is far inferior to EndPoint Permutation for pad generation which he certainly does not employ. Instead he claims that "the dynamic random keys are changed by using random characteristics inherent in the messages themselves." Deriving a random key from messages sent or received or on a separate transmitted channel leaves a vulnerability for cryptanalysts to exploit. It is far safer to generate random keys or subkeys from either a true random source or a high quality prng and employ a master key that sending and receiving devices must use that is kept secret which is not derived

by messages sent or received. The proposed method relies completely on a master key and according to Yanovsky's claim on page 11 he does not use one. On page 12 he also states that there is "no transfer of keys and no master keys needed." On page 17 he repeats again that "The system thus does not depend on the transfer of keys, nor on the existence of a master key." The existence of a master key is an absolute requirement of the proposed protected communication method and system and there is no public description of the EndPoint Permutation procedure as described or used in encryption to secure communications before the patent was applied for in February of 2002. Yanovsky also does not disguise the true binary representation of c-text characters before transmission as the proposed method and system does or use control bits for the express purpose of achieving computational security. For all of the above stated reasons claim 1 should be granted.

Claim 2 on page 5 of the Office Action document is withdrawn from consideration since the preferred binary version will display in the standard format of 64 cipher characters.

Claim 3 and claim 16 on page 5 of the Office Action document should be granted since the synergistic means employed to achieve computational security are unique to the proposed system and method. The claim of Yanovsky, cited for rejecting the proposed claims 3 and 16, would result in a rejected patent claim of any stream cipher made long before or after his patent was granted. Therefore, his claim that was granted had to based upon the unique way his system and method works and application for patent protection

should be granted in this case also, as it was in Yanovsky. The same is true of **Claims 4 and 5** rejected on pages 6 and 7 of the Office Action Summary.

Claim 5 discussed on page 7 should also be granted since the means used to accomplish automatic coding and decoding are superior to Yanovsky. His system requires ongoing real-time synchronization with control bits used to check for data errors and correction as described on pages 3 and 4 in claims 10-13 of Yanovsky. The interjection of redundant bits described is not integral to strengthening the stream cipher as the control bits are in the proposed method. Maintaining device synchronism with interjected bits as Yanovsky describes in claim 20 on page 5 is undesirable. Device independence is a clear advantage of the proposed method since an encrypted message may be read at any time once received and the sending device has no part to play in that process and may even be off at the time. An encrypted message may also be delivered on disk or paper and decrypted after it is entered into a decryption device without the need to ever connect to the device that transmitted the message. It is impossible for the patented protected communication method and system by Yanovsky to do this because of the constant synchronization requirement between sending and receiving devices and the absence of a master key. This tremendous Yanovsky disadvantage makes it clearly inferior to the proposed protected communication method and system.

Claim 6 of the proposed method uses the EndPoint Permutation method to collect digits for character transformation by circling a prime number of elements with a starting point and skip size that varies randomly . On page 7 of the Office Action Summary it states incorrectly that, "Yanovksy discloses a system wherein two numbers are chosen at

random to determine said starting position on said digit ring and said skip size around said digit ring to harvest digits for said character transformation." In reviewing the patent of Yanovsky the use of a prime number of elements arranged in a ring to select digits or bits for summing with plaintext as stream ciphers do is not shown or described. Figure 2 does not show that the function is based upon a prime number of elements. Nor does it show that each prime number of elements can be permuted as it is described in the proposed method. Where does Yanovsky claim that digits are treated as a ring and that only a prime number of elements may be used? Where does he claim in his patent that randomly varying start and skip numbers are used to permute the elements in a ring as it is circled? Where is this shown in figure 2? On page 17 of Yanovsky it says, "It will thus be seen that once two units are initialized between any two parties, cryptographic communication may be conducted between the two parties by means of dynamic random key produced by the outputs of normal state machines at both parties, which are changed internally at the same time in both parties. The system thus does not depend on the transfer of keys, nor on the existence of a master key."

This inferior method of Yanovsky for securing communications is very different from the proposed method in the following ways: 1) No initialization of devices is required in the proposed method. The sending device encrypts independently of the receiving device. The receiving device decrypts independently of the sending parties device. Yanovsky requires that "...cryptographic communication may be conducted between the two parties by means of dynamic random key produced by the outputs of normal state machines at both parties, which are changed internally at the same time in both parties." This requires that both sending and receiving devices are in operation at the same time to change

internally at the same time in both parties to remain in sync with one another. This device dependence requirement is far inferior to the proposed device independence of the proposed method and system. 2) The master key absent in Yanovsky is completely relied upon and necessary in the proposed method and system. 3) The proposed method and system does not employ dynamic random keys "which are changed internally at the same time in both parties" as Yanovsky. The proposed method relies on a master key and randomly varying subkeys that do not require constant monitoring as in Yanovsky to assure that they are the same in the sending and receiving device during transmission. The sending and receiving devices of the proposed method and system operate independently of one another during encryption and decryption. Only one device even needs to be turned on to decrypt or encrypt messages. No real-time synchronization producing keys "that are changed internally at the same time in both parties" is required, as quoted from page 17 of Yanovsky. Yanovsky reinforces again on page 17 that his system requires real-time synchronization when he says, "Cryptographic communication may thus be carried out between each pair of parties by the above-described dynamically-changing random keys so long as the normal state machines of the two parties are synchronized with each other." Yanovsky then describes in great detail on pages 17-29 the cumbersome method of ongoing real-time device synchronization. This labor intensive, complex, and problematic process is undesirable. Perhaps the Yanovsky system works when used in conjunction with other stream ciphers. The proposed steam cipher has no need for a stream cipher management and synchronization device. The proposed communication method and system works fine without it and no successful cryptanalysts attacks have yet been published against the publicly disclosed algorithm.

Claim 6 of the proposed superior method bears no similarity to the claim of Yanovsky and should be granted unless it can be shown that Yanovsky describes a prime number of elements that is circled as elements are selected for harvest using a variable starting point and skip size around the ring and uses those elements to make a pad to disguise p-text.

Page 8 **in reference to claim 7** of the Office Action Summary states "Yanovsky discloses a system wherein said key made from a small amount of text is used to make a multitude of unique digit sequences composed of unique numbers that do not repeat that are used to shuffle digits to achieve said transposition and said key is also expanded to make a large prime string of digits that is relatively random and used as said digit ring to harvest digits used in said transformation (Fig. 2)." Once again, this is not what is described in the patent granted Yanovsky. Where in the patent does Yanovsky describe a large prime string of digits or bits that is involved in transposition and transformation? It is not in the patent. The words "prime" and "ring" are not mentioned once in the patent, nor is the concept. Fig. 2 does not show a prime ring of digits or bits that is circled to harvest the elements it is made up of for permutation as the proposed invention does. Claim 7 in unique and superior to Yanovsky and should be granted.

Page 9 **in reference to claim 8** of the Office Action Summary states "Yanovsky discloses a system wherein the variable digits that determine said starting point on said digit ring and skip size around said digit ring and the variable digits used in said transformation and transposition are mixed into the digit sequence that becomes ciphertext (Fig. 2)." This is

incorrect. In the patent of Yanovsky he does not mention the word "around" once or the word "digits". The bits Yanovsky discusses are used to maintain synchronization between devices as previously discussed and a few bits used for error detection/correction. This cannot be compared to the proposed method which intersperses a much larger number of bits used for a completely different purpose, to achieve computational security. The purpose of the bits in the proposed method and system is to serve as a pointer to a larger number of bits (within a master key) so that a pad can be recreated for decryption, and also to determine the program number used as well as unique sequences used in transposition as well as the true binary representation of the c-text characters. All of these bits deal directly with the encryption process, unlike Yanovsky. The interspersed bits in the proposed method do not serve to maintain device synchronization as in Yanovksy nor do they detect and or correct transmission errors. The interspersed elements, 20 digits in base 10, and around 48 bits in the base 2 algorithm, are integral to the encryption process alone, not error correction or real-time device synchronization.

The proposed claim 8 is unique and superior to Yanovsky and should be granted.

Page 9 **in reference to claim 9** of the Office Action Summary states "Yanovsky discloses a system and method used to code the message is determined by at least one cipher character that is mixed into the ciphertext." On page 3 of Yanovsky it says that "...interjected redundant bits for detecting and/or correcting transmission errors." The bits interspersed into Yanovsky c-text do not have anything to do with the system or method used to encrypt the message as the proposed method does. The interjected bits are for detecting and/or correcting transmission errors which is not a concern of the proposed

method. On page 4 of Yanovsky he further describes the function of his interspersed bits when he says, "...the two parties change their states according to two random bits in the respective segment, and six redundant bits are interjected into the respective segment for detecting and correcting up to two transmission errors in said bits." Yanovsky makes clear that two of the interjected bits are used to change the state of devices. Details of this procedure are in pages 20 and 21 of Yanovsky. The six bits described are for error detection.

The function of the interspersed bits of Yanovksy bears no similarity to the function of bits interspersed in the proposed method. In the proposed method interspersed bits are not used for device synchronization or error correction. They reveal which program in the software was used to encrypt the message received and to reconstruct the pad used to disguise plaintext from the users master key and other elements all having a direct bearing on the encryption process. As stated in claim 9 of the proposed method "the system and method used to code the message is determined by at least one cipher character that is mixed into the ciphertext." The interspersed bits used by Yanovsky are not used to code messages. Yanovsky, instead, says that his patent may use any encryption algorithm on page 16 where he states, "The encrypting algorithm of block E can be any of the known encrypting algorithms." The particular method and system proposed to protect communication is unique and superior to Yanovsky therefore claim 9 should be granted.

Claims 10, 11, 12, 15, 17, and 18 are no longer desired and are withdrawn from consideration.

Page 10 **in reference to claim 13** of the Office Action Summary states "At the time the invention was made, it would have been obvious to a person of ordinary skill in the art to code and decode audio and digital information." Since it "...is the information that is most desirable to secure." The preferred version of the proposed system and method for secure communication and storage of information is binary. It is critical that claim 13 is granted since the proposed method in claim 1 is unique and superior to Yanovsky and others to achieve privacy and it is important to safeguard audio and digital information by cryptography. The One Time Pad (OTP) was not binary at the time of creation but still provided perfect security for text messages. One skilled in the art recognizes that the same encryption means also work in a binary version of OTPs greatly expanding its usefulness. In the same way the base 10 version of the proposed system and method is limited since it provides computational security but is limited to standard keyboard characters. The binary version will allow documents, spreadsheets, images, or any other type of digital information to be encrypted with the same computational security. When patented the binary version will be used to automatically encrypt and decrypt all communications between devices that have been set up with a user key. Correct user keys for decryption will automatically be retrieved from the device receiving communications and used to decrypt once the receiving device recognizes the unique pattern, voice, or other digital sequence associated with each individuals transmission. Claim 13 should be granted since it is obvious that the same unique means are employed to achieve computational security in the base 10 version as the preferred binary version.

Page 10 **in reference to claim 14 and 19** of the Office Action Summary states that, "Yanovsky discloses a system including a variation using a high quality rng (random number generator) used to generate title digits for said start and said skip numbers and other random numbers used in said encryption system and method (Fig. 2)." Yanovksy does not discuss digits used to "start" or "skip" or "jump" around a "prime" ring by these words or any other description around a prime number of bits or digits. On page 16 he says that "The encrypting algorithm of block E can be any of the known encrypting algorithms." The proposed claim is for a unique particular algorithm at the heart of a protected communication method and system not described by Yanovsky that is different and superior in many respects as previously documented that fully relies upon claim 1 as submitted. Claims 14 and 19 are variations dependent upon the unique properties of claim 1 and should be granted.

In reference to claim 16 the proposed method and system variation of claim 1 allows users to use more than one pool of bits to derive random pads from according to the users security needs and desires. So if the standard recommended number of 64,000 bits is deemed inadequate as a user key the user may have as many bit pools of that size as they desire to feel secure. The variation also allows users to combine bits using the XOR function from multiple blocks rather than just one to increase the strength of the encryption. It is important that this variation of claim 1 is granted to not put a limit on the size or randomness of pads generated for transformation of messages using logical operations such as XOR. Claim 16 should be granted.

This is an encryption example using the base ten version of the cipher.

The short string "key" was entered as the user key. "aba" is the plaintext to encrypt. How the 999,983 digit lpms (large prime matrix/string) file is created and the 100 matrices with 100 unique digit pairs each (00-99), found in the mtrcs file, can be seen by examining the algorithm. In the base ten version the 999,983 digits are made relatively random by expanding upon the user key submitted. The 100 matrices are simply 100 unique strings of 100 elements each of the elements 00-99 with each element being unique in the set. In other words each set would only have one number 17 within it, for example. These can be seen with the algorithm running on a computer.

Open the software program and enter a key. Once this is done, and with the program running, search for the "lpms" and "mtrcs" files on your computer and open to see them with WordPad or Notepad. These digit tables are used as subkeys in the encryption and decryption process and are deleted when the program stops running.

Since the base ten version of the cipher is made up of ten separate programs one is chosen at random by the program for encryption. The ten programs vary in the matrices used in character assignments and shuffling.

BASE TEN ENCRYPTION EXAMPLE:

Program 2 was automatically chosen at random in this example.

start = 900904 Chosen at random. This number determines the starting point on the lpms ring.

jump = 015749 Chosen at random. This number determines the jump size around the lpms when treated as a ring of digits. Each time it lands a copy of that digit is harvested from the ring for a maximum collection size of 10,000 digits per encryption block.

The plaintext "aba" is converted into a digit string. Each p-text character is converted into a digit pair according to the program number used and one of the 100 matrices. In this case a=37 and b=04. A different program would have different digit pair assignments.

370437 is "aba" after conversion to digits.

All valid user key characters have a unique three digit assignment with the max size of a user key of 4,000 characters. All the user key characters are converted into their three digit numbers and Chinese summed (addition without carries) to get a single three digit number. This Key Value is used to further corrupt the entire message.

The Key Value is 467 in this example. The number is then multiplied as many times as needed to be the same length as the p-text string after conversion to digits and used to corrupt an entire message if one key character is wrong.

370437 P-text after conversion to digits

467467 + Key Value using Chinese Addition (No carries)

737894 = (P-text + KeyValue)

In this example the following digits were harvested from the lpms. Enough digits are harvested to cover the P-text digits in groups of 100.

HarvestedString =

515990042376135084121714272062191792330644728040181695593442949494206381

171895181774187094047196086 6

The string of harvested digits is then shuffled using one of the 100 matrices with that number being determined by the program number.

ShuffledHarvestedString =

161128289293841948507104971155016467482541947751270900600953241147018122

148494807093736346261349799 6

The six digits needed to match the message length, 161128 in this case, are Chinese summed (addition without carries) with the digit string. Digits not needed are discarded. This number is then summed with the previous string.

737894 (P-text + KeyValue)

161128 + (ShuffledHarvestedString)

898912 = (P-text +Key Value + ShuffledHarvestedString)

Two more digit pairs are chosen at random for modifying matrices before they are used to reorder digit strings. The digit pairs randomly chosen are 69 and 77 in this case.

Another shuffle of the resulting digit string is done using matrices determined by program number and modified by a digit pair chosen at random. As each digit pair in a

matrix is Chinese Summed with a digit pair at random it is transformed and the resulting 100 pairs remain unique in the matrix.

898912 Before Shuffle

198928 After Shuffle by modified matrix

Now the controlling digits are interspersed into the message string in this order: Start number, Jump number, 1st digit pair, 2nd digit pair. In this case the digits are 9009040157496977. They are interspersed in the message string in positions determined by the lpms with the result after each one is added shown below. The control digits are well mixed in with message digits of any length as they are in this example.

1989928

19809928

198099028

1989099028

10989099028

109489099028

1009489099028

10094890990128

100948950990128

1009489509901728

10094849509901728

100948495909901728

1009694849590979017728

The resulting mix of 22 digits is now converted into Cipher Characters as determined by program number and one of the matrices.

1009694849590979017728 (22 digit message & control digit mix)

?y÷5HCynbD: (11 Cipher-Character message)

Another random digit pair is chosen. 33 in this case. A second program character of 35 is chosen for a difference of 2 which is the program number.

?y÷5HCynbGD: The character "G" representing 33 in this case is interspersed.

?y÷I5HCynbGD: The character "I" representing 35 in this case is interspersed which completes the encryption process of the plaintext "aba" to the c-text "?y÷I5HCynbGD:"

Note that the c-text is 10 characters longer than the p-text. In the detailed example 3 p-text characters becomes 13 as c-text. Ten extra characters will be in each block of c-text up to 5,010 c-text characters. Encrypting 5,000 characters of p-text will result in a c-text message of 5,010 characters. 10,000 p-text characters will encrypt as 10,020 c-text characters, and so forth.

Obstacles that cryptanalysts must overcome:

The base two system has 64 possible c-text characters. The following short c-text message "**JUgohwwVHsD/2ixDq8+t**" has 20 characters or 120 transmitted bits. One of the twenty characters in this example is the anchor character. It is the only character that will always be in the same numbered position. The remaining 19 characters will usually be a mixture of control bits and disguised p-text bits. Regardless of what is coded the c-text will always be statistically random.

1) the cryptanalyst should determine which one character is the anchor character. This is the only character that will always be in the same numbered position. It will vary randomly to be any of the 64 possible c-text characters. In this short message the cryptanalyst has a 1:20 chance of guessing correctly.

2) the cryptanalyst should find what the remaining 19 c-text characters represent in binary form. The binary form each has varies according to the value of the anchor character and the master key. The true binary value is not the same value that is transmitted to the receiving device. So if the first "w" in the c-text string is the anchor character the next "w" may be "101100." The remaining characters would each have an assigned true binary value. This value is determined by the user key and program number. The chances of guessing each true binary value correctly is not good at 1:64 factorial or 1:1.269 x 10^89 A difficult guess.

3) the cryptanalyst should remove the anchor character and find and remove the six control bits that are used with it to determine the program number. The numbered positions of the six bits are determined by the master key and their position in the c-text varies according to the random anchor character and master key. The six bits are removed from the bit sequence and the difference in value between them and the anchor character reveal what program number was used to encrypt the c-text. The odds of finding the correct 6 bits from the string of 114 bits after the anchor is removed are not good.

4) the cryptanalyst should separate the remaining control bits from the disguised p-text bits. Out of the remaining 108 bits they should pick which 60 are the disguised p-text. Once again the odds of doing so are not good.

5) the cryptanalyst should then find the correct order of the 60 disguised p-text bits since their order was rearranged using EndPoint Permutation. Odds of guessing are 1:3,540 for this short message if a single round of EndPoint Permutation was used.

6) the cryptanalyst should find what pool of bits was drawn from to form the pad for disguising p-text since more than one bit pool of 60,000 bits may exist.

7) the cryptanalyst should find what 60 bits from the bit pool chosen were XOR'd with the 60 p-text bits to disguise it. Odds of guessing are 2^60 or 1.53 x 10^18.

8) the cryptanalyst should find the first unique binary sequence that is 384 bits long and the 63 variants of it since they are used to sum with the p-text binary sequence before EndPoint Permutation is used. In this case only the first 60 bits are needed. Odds of guessing are 1.53 x 10^18.

These are just a few of the obstacles that cryptanalysts will find insurmountable regardless of the amount of computer power applied. The numbers are too large for them to succeed. Schneier in Applied Cryptography, 2nd Edition on page 239 says, "The complexity of a brute force attack against a 56 bit key is 2^56; against a 112-bit key the complexity is 2^112. The former is possible; the latter isn't." As demonstrated from the

numbers above the difficulty of cracking the proposed method and system the number 2^112 or 5.19 x 10^33, said to be impossible to break, is easily surpassed.

Statistical analysis of c-text will also not help since it will always be random when the proposed method and system is used as recommended. The protected communication method and system will endure the following types of attacks described by Schneier in Applied Cryptography, 2nd Edition on pages 5-6.: 1)ciphertext-only attack 2) known-plaintext attack 3) chosen-plaintext attack and 4) adaptive-chosen-plaintext attack.

In summary:

On page 16 of the Yanovsky patent it states, "The encrypting algorithm of block E can be any of the known encrypting algorithms." This makes it clear that his system is designed to work in conjunction with an encryption algorithm while the proposed system and method for secure communication is itself an encrypting algorithm, a strengthened stream cipher, that does not need the Yanovsky system for error control or correction or device synchronization.

The proposed protected communication method and system is unique, computationally secure, easy to use. The following action should be taken:

Claim 1 should be granted
Claim 2 has been withdrawn
Claims 3,4,5,6,7,8, and 9 should be granted
Claims 10-12 have been withdrawn
Claims 13 and 14 should be granted
Claim 15 is withdrawn
Claim 16 should be granted
Claims 17 and 18 are withdrawn
Claim 19 should be granted

c-text randomness compared to π

pi, 1st 5k digits	1	2	3	4	5	6	7
0	466	32	5	0			
1	531	50	5	0			
2	496	44	6	1	0		
3	460	37	2	0			
4	508	46	7	0			
5	525	41	5	0			
6	513	47	4	0			
7	488	44	2	1	0		
8	492	38	2	1	0		
9	521	47	3	1	1	1	0

c-text, encrypted the 1st 2,490 digits of pi and converted to 5k digits

0	504	48	3	0		
1	500	45	10	1	0	
2	476	41	5	0		
3	493	38	6	0		
4	507	49	5	2	0	
5	522	36	2	0		
6	457	41	7	0		
7	500	49	5	1	0	
8	530	57	3	1	0	
9	511	36	3	0		

c-text, encrypted 2,490 A's and converted c-text to 5k digits
All on this page encrypted with same random 4k c-text key dated 11-13-05.

0	466	37	3	1	0	
1	513	43	7	0		
2	506	45	2	0		
3	482	38	4	0		
4	458	36	4	1	1	0
5	516	52	2	0		
6	517	58	5	0		
7	511	50	2	0		
8	493	47	5	1	0	
9	538	48	7	2	2	0

Same all A's encrypted as above with same key

0	518	47	5	0		
1	525	44	2	1	0	
2	496	49	6	1	0	
3	516	44	5	1	0	
4	457	47	6	0		
5	503	40	4	0		
6	512	60	4	2	1	0
7	485	47	2	0		
8	473	43	7	1	0	
9	515	46	6	0		